OBJECT ORIENTED PROGRAMMING WITH PYTHON

Second Class

1st Semester

- Tuples
- Unpacking
- Dictionarires
- Functions
- Parameters
- Keyword Arguments
- Return Statement

TUPLES

- Tuples are similar to list, so we can use them to store a list of items but unlike lists we can't modify them (can't add new items, can't remove existing item), thus tuples is immutable.

```
Numbers=(1,2,3)
Numbers.
```

Here the methods related to tuples will be listed (**count** and **index**)

count to count the number of occurrence of an item

Index to find the index of the first occurrence of an item

```
Numbers=(1,2,3)
Numbers[0]=10
print(Numbers[0])
```

UNPACKING

A powerful feature in Python coding

```
coordinateds=(1,2,3)
 coordinateds[0] * coordinateds[1] * coordinateds[2]
```

```
X= coordinateds[0]
Y= coordinateds[1]
Z= coordinateds[2]
```

In Python we can use unpacking feature to unpack list to variables

```
X,Y,Z= coordinateds
```

Note:
This is not limited to Tuples it's also can be used with lists

DICTIONATRIES

Use dictionary in situations were we want to store information that come at **key-value pair**

Example:

Think of a customer, a customer has attributes (information) like name, Email, phone, address and so on

Each of this attribute has a value key like

name : Jonh Smith

Email : john@gmail.com

Phone : 008645876

Here we have a bunch of key-value pairs

Now using the Dictionary we can store key-value pairs

DICTIONATRIES

```
Customer= {
    “name” : “Jonh Smith”,
    “ Email” : “john@gmail.com”,
    “age” :  30,
    “Is_verified”=True
}
```

We can reach any item using square brackets [ ]

print (customer[“name”]) #Returns **John Smith**

print (customer[“birthdate“]) #Returns error message

print (customer[“Name“]) #Returns error message

DICTIONATRIES

To solve the error problem we can use **get()**

print (customer.get("Name")) #Returns **None**

Note: **None** is an object represents the absence of value.

Also we can use the **default value** like

print (customer.get("birthdate","Jan 1 1980") #Returns Jan 1 1980

To update the key value

customer["name"] = "Jack Smith"

To add a new key

customer["birthdate"] = "Jan 1 1980"

EXERCISE

Write a program that translates the numbers(digits) into words like

Phone: 1234

One Two Three Four

Solution

```
phone=input(“Phone: “)
digits_mapping={
    “1” : ”One”,
    “2” : “Two”
    “3” :”Three”
    “4”: “Four”
}
output=“”
for ch in phone:
    output+= digits_mapping.get(ch,”!”) +” “
print(output)
```

Phone: 12345

One Two Three Four !

EMOJI CONVERTER

Write an application that maps characters :) to ☺ and :(to ☹

Solution

```
message=input("> ")
words = message.split(' ')
emojis = {

    ":)":"☺",

    ":(":" ☹ "
}
output =‘ ‘
for word in words:
    print(word)
    output+=emojis.get(word,word)+' '
print(output)
```

I am happy :) ---> I am happy ☺
I am sad): ---> I am sad ☹

FUNCTIONS

The function is the better way to organize our code, we sometimes need to breakup our code into smaller, manageable and more maintainable chunks.

When building large complex programs we should break up out code into smaller reusable chunks which we call function to better organize our code.

Let us write a simple program for printing a greeting messages

```
print ('Hi there!')
print(' Welcome to our class')
```

If we need these printing in another programs, we can put them in function that we can reuse.